GUIA 8 DE LENGUAJES: BATALLAS ENTRE PARADIGMAS, TESIS DE CHURCH

En esta guia compararemos los tres paradigmas de computabilidad efectiva que hemos desarrollado anteriormente. Para esto probaremos que cada uno de dichos paradigmas "vence" al otro en el sentido que incluye por lo menos todas las funciones que incluye el otro en su modelizacion del concepto de funcion Σ-efectivamente computable. Por supuesto, esto dice que los tres son equivalentes.

Neumann vence a Godel

Usando macros podemos ahora probar que el paradigma imperativo de Neumann es por lo menos tan abarcativo como el funcional de Godel. Mas concretamente:

Teorema 1 (Neumann vence a Godel)**.** *Si h es Σ-recursiva, entonces h es Σ-computable.*

Proof. Probaremos por induccion en k que

(*) Si $h \in \mathrm{R}_k^\Sigma$, entonces h es Σ-computable.

El caso $k = 0$ es dejado al lector. Supongamos (*) vale para k, veremos que vale para $k+1$. Sea $h \in \mathrm{R}_{k+1}^\Sigma - \mathrm{R}_k^\Sigma$. Hay varios casos

Caso 1. Supongamos $h = M(P)$, con $P : \omega \times \omega^n \times \Sigma^{*m} \to \omega$, un predicado perteneciente a R_k^Σ. Por hipotesis inductiva, P es Σ-computable y por lo tanto tenemos un macro

$$\left[\text{IF } P(\mathrm{V}1, ..., \mathrm{V}\overline{n+1}, \mathrm{W}1, ..., \mathrm{W}\bar{m}) \text{ GOTO A1}\right]$$

lo cual nos permite realizar el siguiente programa

$$\begin{array}{ll} \mathrm{L2} & \left[\text{IF } P(\mathrm{N}\overline{n+1}, \mathrm{N}1, ..., \mathrm{N}\bar{n}, \mathrm{P}1, ..., \mathrm{P}\bar{m}) \text{ GOTO L1}\right] \\ & \mathrm{N}\overline{n+1} \leftarrow \mathrm{N}\overline{n+1} + 1 \\ & \text{GOTO L2} \\ \mathrm{L1} & \mathrm{N}1 \leftarrow \mathrm{N}\overline{n+1} \end{array}$$

Es facil chequear que este programa computa h.

Caso 2. Supongamos $h = R(f, \mathcal{G})$, con

$$f : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \to \Sigma^*$$
$$\mathcal{G}_a : S_1 \times ... \times S_n \times L_1 \times ... \times L_m \times \Sigma^* \times \Sigma^* \to \Sigma^*,\ a \in \Sigma$$

elementos de R_k^Σ. Sea $\Sigma = \{a_1, ..., a_r\}$. Por hipotesis inductiva, las funciones f, $\mathcal{G}_a$, $a \in \Sigma$, son Σ-computables y por lo tanto tenemos macros

$$\left[\mathrm{W}\overline{m+1} \leftarrow f(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m})\right]$$

$$\left[\mathrm{W}\overline{m+3} \leftarrow \mathcal{G}_{a_i}(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m}, \mathrm{W}\overline{m+1}, \mathrm{W}\overline{m+2})\right],\ i = 1, ..., r$$

Podemos entonces hacer el siguiente programa:

$$
\begin{array}{ll}
 & \left[\mathrm{P}\overline{m+3} \leftarrow f(\mathrm{N}1, ..., \mathrm{N}\bar{n}, \mathrm{P}1, ..., \mathrm{P}\bar{m})\right] \\
\mathrm{L}\overline{r+1} & \text{IF } \mathrm{P}\overline{m+1} \text{ BEGINS } a_1 \text{ GOTO L1} \\
 & \qquad \vdots \\
 & \text{IF } \mathrm{P}\overline{m+1} \text{ BEGINS } a_r \text{ GOTO L}\bar{r} \\
 & \text{GOTO L}\overline{r+2} \\
\mathrm{L}1 & \mathrm{P}\overline{m+1} \leftarrow \ \curvearrowright \mathrm{P}\overline{m+1} \\
 & \left[\mathrm{P}\overline{m+3} \leftarrow \mathcal{G}_{a_1}(\mathrm{N}1, ..., \mathrm{N}\bar{n}, \mathrm{P}1, ..., \mathrm{P}\bar{m}, \mathrm{P}\overline{m+2}, \mathrm{P}\overline{m+3})\right] \\
 & \mathrm{P}\overline{m+2} \leftarrow \mathrm{P}\overline{m+2}.a_1 \\
 & \text{GOTO L}\overline{r+1} \\
 & \qquad \vdots \\
\mathrm{L}\bar{r} & \mathrm{P}\overline{m+1} \leftarrow \ \curvearrowright \mathrm{P}\overline{m+1} \\
 & \left[\mathrm{P}\overline{m+3} \leftarrow \mathcal{G}_{a_r}(\mathrm{N}1, ..., \mathrm{N}\bar{n}, \mathrm{P}1, ..., \mathrm{P}\bar{m}, \mathrm{P}\overline{m+2}, \mathrm{P}\overline{m+3})\right] \\
 & \mathrm{P}\overline{m+2} \leftarrow \mathrm{P}\overline{m+2}.a_r \\
 & \text{GOTO L}\overline{r+1} \\
\mathrm{L}\overline{r+2} & \mathrm{P}1 \leftarrow \mathrm{P}\overline{m+3}
\end{array}
$$

Es facil chequear que este programa computa h.

El resto de los casos son dejados al lector. ■ Un corolario importante de esta batalla es el:

Proposición 2 (Segundo Manatial de Macros)**.** *Sea Σ un alfabeto finito. Si*

$$f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \omega$$
$$g : D_g \subseteq \omega^n \times \Sigma^{*m} \to \Sigma^*$$
$$P : D_P \subseteq \omega^n \times \Sigma^{*m} \to \{0, 1\}$$

son Σ-recursivas, entonces en $\mathcal{S}^\Sigma$ hay macros

$$\left[\mathrm{V}\overline{n+1} \leftarrow f(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m})\right]$$
$$\left[\mathrm{W}\overline{m+1} \leftarrow g(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m})\right]$$
$$[\text{IF } P(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m}) \text{ GOTO A1}]$$

Recordemos que el Primer Manantial de Macros es dado en la Guia 7 al final de la seccion sobre macros.

Ejercicio 1: Pruebe el Segundo Manantial de Macros.

Se lleno de macros. Cabe destacar que el Segundo Manantial de Macros nos dice que en $\mathcal{S}^\Sigma$ hay macros

$$\left[\mathrm{V}\overline{n+1} \leftarrow f(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m})\right]$$
$$\left[\mathrm{W}\overline{m+1} \leftarrow g(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m})\right]$$
$$[\text{IF } P(\mathrm{V}1, ..., \mathrm{V}\bar{n}, \mathrm{W}1, ..., \mathrm{W}\bar{m}) \text{ GOTO A1}]$$

para todas las funciones Σ-mixtas y predicados Σ-mixtos que hemos trabajado hasta el momento en la materia ya que todas eran Σ-p.r.. Esto fortalece mucho al lenguaje $\mathcal{S}^\Sigma$ ya que ahora tenemos macros para todas las funciones y predicados cotidianos en la matematica, ademas de tener macros para todas las funciones y

predicados Σ-computables, debido al Primer Manantial de Macros. Veamos un ejemplo:

Lema 3. *Supongamos $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ son conjuntos Σ-enumerables. Entonces $S_1 \cup S_2$ es Σ-enumerable.*

Proof. Podemos suponer que ni S_1 ni S_2 son vacios ya que de lo contrario el resultado es trivial. Ademas supondremos que $n = 2$ y $m = 1$.

La idea de la prueba es la misma que la que usamos para probar que la union de conjuntos Σ-efectivamente enumerables es Σ-efectivamente enumerable. Daremos usando macros un programa que enumera a $S_1 \cup S_2$ y luego aplicaremos la Proposicion de Caracterizacion de Σ-enumerabilidad de la Guia 7. Por hipotesis hay funciones $F : \omega \to \omega \times \omega \times \Sigma^*$ y $G : \omega \to \omega \times \omega \times \Sigma^*$ tales que $F_{(1)}$, $F_{(2)}$, $F_{(3)}$, $G_{(1)}$, $G_{(2)}$ y $G_{(3)}$ son Σ-computables, $\mathrm{Im}(F) = S_1$ y $\mathrm{Im}(G) = S_2$. O sea que nuestro Primer Manantial de Macros nos dice que en $\mathcal{S}^\Sigma$ hay macros

$$\begin{gathered}
\big[\mathrm{V2} \leftarrow F_{(1)}(\mathrm{V1})\big] \\
\big[\mathrm{V2} \leftarrow F_{(2)}(\mathrm{V1})\big] \\
\big[\mathrm{W1} \leftarrow F_{(3)}(\mathrm{V1})\big] \\
\big[\mathrm{V2} \leftarrow G_{(1)}(\mathrm{V1})\big] \\
\big[\mathrm{V2} \leftarrow G_{(2)}(\mathrm{V1})\big] \\
\big[\mathrm{W1} \leftarrow G_{(3)}(\mathrm{V1})\big]
\end{gathered}$$

Ya que el predicado $Par = \lambda x[x \text{ es par}]$ es Σ-p.r., el Segundo Manantial de Macros nos dice que en $\mathcal{S}^\Sigma$ hay un macro:

$$[\text{IF } Par(\mathrm{V1}) \text{ GOTO A1}]$$

el cual escribiremos de la siguiente manera mas intuitiva

$$[\text{IF V1 es par GOTO A1}]$$

Ya que la funcion $D = \lambda x[\lfloor x/2 \rfloor]$ es Σ-p.r., el Segundo Manantial de Macros nos dice que hay un macro:

$$[\mathrm{V2} \leftarrow D(\mathrm{V1})]$$

el cual escribiremos de la siguiente manera mas intuitiva

$$[\mathrm{V2} \leftarrow \lfloor \mathrm{V1}/2 \rfloor]$$

Sea $\mathcal{P}$ el siguiente programa:

$$\begin{array}{ll}
 & [\text{IF N1 es par GOTO L1} \\
 & \mathrm{N1} \leftarrow \mathrm{N1} \dot{-} 1 \\
 & [\mathrm{N1111} \leftarrow \lfloor \mathrm{N1}/2 \rfloor] \\
 & \big[\mathrm{N1} \leftarrow G_{(1)}(\mathrm{N1111})\big] \\
 & \big[\mathrm{N2} \leftarrow G_{(2)}(\mathrm{N1111})\big] \\
 & \big[\mathrm{P1} \leftarrow G_{(3)}(\mathrm{N1111})\big] \\
 & \text{GOTO L2} \\
\mathrm{L1} & [\mathrm{N1111} \leftarrow \lfloor \mathrm{N1}/2 \rfloor] \\
 & \big[\mathrm{N1} \leftarrow F_{(1)}(\mathrm{N1111})\big] \\
 & \big[\mathrm{N2} \leftarrow F_{(2)}(\mathrm{N1111})\big] \\
 & \big[\mathrm{P1} \leftarrow F_{(3)}(\mathrm{N1111})\big] \\
\mathrm{L2} & \text{SKIP}
\end{array}$$

Es facil ver que $\mathcal{P}$ cumple (a) y (b) de (2) de la Proposicion de Caracterizacion de Σ-enumerabilidad de la Guia 7 por lo cual $S_1 \cup S_2$ es Σ-enumerable. ■

En forma analoga a lo hecho recien, se pueden probar, copiando la respectiva idea en el paradigma de la computabilidad efectiva, los siguientes resultados, los cuales son dejados como ejercicios.

Ejercicio 2: Use los macros asociados a las "bajadas" y a $*^{\leq}$ para dar un programa que enumere a $\omega \times \omega \times \Sigma^*$, es decir que cumpla (a) y (b) de (2) de la Proposicion de Caracterizacion de Σ-enumerabilidad de la Guia 7.

Ejercicio 3: Supongamos $S_1, S_2 \subseteq \omega^n \times \Sigma^{*m}$ son conjuntos Σ-enumerables. Entonces $S_1 \cap S_2$ es Σ-enumerable. (Hacer el caso $n = 2$, $m = 1$ e inspirese en el paradigma efectivo)

En la Guia 3 probamos que si $S \subseteq \omega^n \times \Sigma^{*m}$ es Σ-efectivamente computable entonces S es Σ-efectivamente enumerable. Via el uso de macros adecuados podemos copiar la idea de la prueba de dicho resultado para probar el siguiente

Lema 4. *Sea $S \subseteq \omega^n \times \Sigma^{*m}$. Si S es Σ-computable, entonces S es Σ-enumerable*

Ejercicio 4: Pruebe el lema anterior (haga el caso $n = 2$, $m = 1$)

Ejercicio 5: Si $S \subseteq \Sigma^*$ es Σ-enumerable, entonces

$$T = \{\alpha \in \Sigma^* : \text{existe } \beta \in S \text{ tal que } \alpha \text{ es subpalabra de } \beta\}$$

también es Σ-enumerable.

Ejercicio 6: Sean $S \subseteq \omega$ y $L \subseteq \{@, \uparrow\}^*$ tales que $(0, \varepsilon) \in S \times L$. Entonces $S \times L$ es $\{@, \uparrow\}$-enumerable si y solo si ambos S y L son $\{@, \uparrow\}$-enumerables

O sea que con el uso de nuestros poderosos macros asociados a funciones y predicados Σ-p.r. (gracias al Segundo Manatial) mas los que provee el Primer Manatial podemos simular los procedimientos efectivos realizados dentro del paradigma filosofico con programas concretos del lenguaje $\mathcal{S}^{\Sigma}$. Pero esto no es del todo asi, ya que los ejemplos vistos recien no hacen uso del concepto de "ejecucion de una cantidad de pasos", idea que en muchos diseños de nuestros procedimentos efectivos es usada. Por ejemplo si queremos "copiar" dentro del paradigama imperativo la prueba del resultado

- Si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to \omega$ es una funcion Σ-efectivamente computable, entonces D_f es Σ-efectivamente enumerable

notaremos la dificultad de aun no poder hablar de cantidad de pasos en la ejecucion del programa que compute a f. O sea nuestro Primer Manatial nos da un macro de asignacion para f pero no es claro de como correr una expansion de dicho macro una cierta cantidad de veces. Esto lo lograremos usando macros asociados a las funciones $Halt^{n,m}$, $E^{n,m}_{\#}$ y $E^{n,m}_{*}$ las cuales se usan en la batalla Godel vence a Neumann que viene a continuacion. Estas funciones seran clave a la hora de simular con programas a procedimientos efectivos que en su funcionamiento involucran el funcionamiento de otros procedimientos efectivos.

O sea que luego de ambas batallas entre Godel y Neumann, el paradigma imperativo se vera fortalecido sustancialmente. Tambien mas adelante en la Guia 9

veremos como los desarrollos hechos en ambas batallas entre Godel y Neumann fortalecen al paradigma funcional.

Godel vence a Neumann

Primero definiremos tres funciones las cuales contienen toda la informacion acerca del funcionamiento del lenguaje $\mathcal{S}^{\Sigma}$. Sean $n, m \in \omega$, fijos. Definamos

$$i^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \omega$$
$$E_{\#}^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \omega^{[\mathbf{N}]}$$
$$E_{*}^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \Sigma^{*[\mathbf{N}]}$$

de la siguiente manera

$$(i^{n,m}(0,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#}^{n,m}(0,\vec{x},\vec{\alpha},\mathcal{P}), E_{*}^{n,m}(0,\vec{x},\vec{\alpha},\mathcal{P})) = (1,(x_1,...,x_n,0,...),(\alpha_1,...,\alpha_m,\varepsilon,...))$$
$$(i^{n,m}(t+1,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#}^{n,m}(t+1,\vec{x},\vec{\alpha},\mathcal{P}), E_{*}^{n,m}(t+1,\vec{x},\vec{\alpha},\mathcal{P})) =$$
$$=$$
$$S_{\mathcal{P}}(i^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}))$$

Notese que

$$(i^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}))$$

es la descripcion instantanea que se obtiene luego de correr $\mathcal{P}$ una cantidad t de pasos partiendo del estado

$$((x_1,...,x_n,0,...),(\alpha_1,...,\alpha_m,\varepsilon,...))$$

Es importante notar que si bien $i^{n,m}$ es una funcion $(\Sigma \cup \Sigma_p)$-mixta, ni $E_{\#}^{n,m}$ ni $E_{*}^{n,m}$ lo son.

Definamos para cada $j \in \mathbf{N}$, funciones

$$E_{\#j}^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \omega$$
$$E_{*j}^{n,m} : \omega \times \omega^n \times \Sigma^{*m} \times \mathrm{Pro}^{\Sigma} \to \Sigma^{*}$$

de la siguiente manera

$$E_{\#j}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) = j\text{-esima coordenada de } E_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P})$$
$$E_{*j}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) = j\text{-esima coordenada de } E_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P})$$

(es claro que estas funciones son $(\Sigma \cup \Sigma_p)$-mixtas). Notese que

$$E_{\#}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) = (E_{\#1}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_{\#2}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}),...)$$
$$E_{*}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}) = (E_{*1}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}), E_{*2}^{n,m}(t,\vec{x},\vec{\alpha},\mathcal{P}),...)$$

Aceptaremos sin prueba la siguiente proposicion (ver el apunte por una prueba).

Proposición 5. *Sean $n, m \geq 0$. Las funciones $i^{n,m}$, $E_{\#j}^{n,m}$, $E_{*j}^{n,m}$, $j = 1, 2, ...$, son $(\Sigma \cup \Sigma_p)$-p.r.*

Las funciones $Halt^{n,m}$ y $T^{n,m}$. Dados $n, m \in \omega$, definamos:

$$Halt^{n,m} = \lambda t\vec{x}\vec{\alpha}\mathcal{P}\left[i^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}) = n(\mathcal{P}) + 1\right]$$

Notese que $D_{Halt^{n,m}} = \omega \times \omega^n \times \Sigma^{*m} \times \text{Pro}^{\Sigma}$ (ojo que aqui la notacion lambda es respecto del alfabeto $\Sigma \cup \Sigma_p$). Ademas notese que usamos la variable $\mathcal{P}$ en la notacion lambda por un tema de comodidad psicologica dado que $i^{n,m}$ esta definida solo cuando la ultima coordenada es un programa pero podriamos haber escrito $\lambda t\vec{x}\vec{\alpha}\alpha\left[i^{n,m}(t, \vec{x}, \vec{\alpha}, \alpha) = n(\alpha) + 1\right]$ y sigue siendo la misma funcion.

Cabe destacar que $Halt^{n,m}$ tiene una descripcion muy intuitiva, ya que dado $(t, \vec{x}, \vec{\alpha}, \mathcal{P}) \in \omega \times \omega^n \times \Sigma^{*m} \times \text{Pro}^{\Sigma}$, tenemos que $Halt^{n,m}(t, \vec{x}, \vec{\alpha}, \mathcal{P}) = 1$ si y solo si el programa $\mathcal{P}$ se detiene luego de t pasos partiendo desde el estado $\|x_1, ..., x_n, \alpha_1, ..., \alpha_m\|$.

Aceptaremos sin demostracion el siguiente lema (ver el apunte por una prueba).

Lema 6. (a) Pro^{Σ} *es un conjunto* $(\Sigma \cup \Sigma_p)$*-p.r.*
(b) $\lambda\mathcal{P}\left[n(\mathcal{P})\right]$ *y* $\lambda i\mathcal{P}\left[I_i^{\mathcal{P}}\right]$ *son funciones* $(\Sigma \cup \Sigma_p)$*-p.r..*

Ejercicio 7: De una funcion $f : \omega \times \text{Pro}^{\Sigma} \to \text{Pro}^{\Sigma}$ la cual sea $(\Sigma \cup \Sigma_p)$-p.r. y cumpla que
(a) $D_{\Psi^{0,1,\#}_{f(x,\mathcal{P})}} = D_{\Psi^{0,1,\#}_{\mathcal{P}}}$, para cada $x \in \omega$ y $\mathcal{P} \in \text{Pro}^{\Sigma}$
(b) $\Psi^{0,1,\#}_{f(x,\mathcal{P})}(\alpha) = \Psi^{0,1,\#}_{\mathcal{P}}(\alpha) + x$, para cada $x \in \omega$, $\mathcal{P} \in \text{Pro}^{\Sigma}$ y $\alpha \in D_{\Psi^{0,1,\#}_{\mathcal{P}}}$
Pruebe que f es $(\Sigma \cup \Sigma_p)$-p.r.

Ahora podemos probar el siguiente importante resultado.

Lema 7. $Halt^{n,m}$ *es* $(\Sigma \cup \Sigma_p)$*-p.r.*

Proof. Notar que $Halt^{n,m} = \lambda xy[x = y] \circ \left[i^{n,m}, \lambda\mathcal{P}[n(\mathcal{P}) + 1] \circ p^{1+n,m+1}_{1+n+m+1}\right]$. ■

Ahora definamos $T^{n,m} = M(Halt^{n,m})$. Notese que

$$D_{T^{n,m}} = \{(\vec{x}, \vec{\alpha}, \mathcal{P}) : \mathcal{P} \text{ se detiene partiendo de } \|x_1, ..., x_n, \alpha_1, ..., \alpha_m\|\}$$

y para $(\vec{x}, \vec{\alpha}, \mathcal{P}) \in D_{T^{n,m}}$ tenemos que $T^{n,m}(\vec{x}, \vec{\alpha}, \mathcal{P}) =$ cantidad de pasos necesarios para que $\mathcal{P}$ se detenga partiendo de $\|x_1, ..., x_n, \alpha_1, ..., \alpha_m\|$. En algun sentido, la funcion $T^{n,m}$ mide el "tiempo" que tarda en detenerse $\mathcal{P}$ y de ahi su nombre.

Proposición 8. $T^{n,m}$ *es* $(\Sigma \cup \Sigma_p)$*-recursiva*

Proof. Es directo del lema de minimizacion ya que $Halt^{n,m}$ es $(\Sigma \cup \Sigma_p)$-p.r. ■

Ahora nos sera facil probar que el paradigma de Godel es por lo menos tan abarcativo como el imperativo de Von Neumann. Mas concretamente:

Teorema 9 (Godel vence a Neumann)**.** *Si* $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ *es* Σ*-computable, entonces* f *es* Σ*-recursiva.*

Proof. Haremos el caso $O = \Sigma^*$. Sea $\mathcal{P}_0$ un programa que compute a f. Primero veremos que f es $(\Sigma \cup \Sigma_p)$-recursiva. Note que

$$f = E^{n,m}_{*1} \circ \left[T^{n,m} \circ \left[p^{n,m}_1, ..., p^{n,m}_{n+m}, C^{n,m}_{\mathcal{P}_0}\right], p^{n,m}_1, ..., p^{n,m}_{n+m}, C^{n,m}_{\mathcal{P}_0}\right]$$

donde cabe destacar que $p^{n,m}_1, ..., p^{n,m}_{n+m}$ son las proyecciones respecto del alfabeto $\Sigma \cup \Sigma_p$, es decir que tienen dominio $\omega^n \times (\Sigma \cup \Sigma_p)^{*m}$. Esto nos dice que f es

$(\Sigma \cup \Sigma_p)$-recursiva. O sea que el Teorema de Independencia del Alfabeto nos dice que f es Σ-recursiva. ■

Aceptaremos sin prueba la siguiente proposicion.

Proposición 10. *La funcion $T^{n,m}$ no es $(\Sigma \cup \Sigma_p)$-p.r.*

Corolario 11. *La minimizacion de un predicado Σ-p.r. no necesariamente es Σ-p.r.*

Proof. Por definicion $T^{n,m} = M(Halt^{n,m})$. ■

Uso de macros asociados a las funciones $Halt^{n,m}$, $E^{n,m}_{\#}$ y $E^{n,m}_{*}$. Aqui veremos, con ejemplos, como ciertos macros nos permitiran dentro de un programa hablar acerca del funcionamiento de otro programa. En este sentido los desarrollos de las dos batallas entre Neumann y Godel nos permiten fortalecer notablemente al paradigma imperativo en su roll modelizador (o simulador) de los procedimientos efectivos. Esto es importante ya que el paradigma mas comodo, amplio e intuitivo es sin duda el filosofico de Leibniz.

Veamos el primer ejemplo. Probaremos que:

- Si $f : D_f \subseteq \omega \to \omega$ es Σ-computable, $0 \in D_f$ y $f(0) = 2$, entonces

$$S = \{x \in D_f : f(x) \neq 0\}$$

es Σ-enumerable.

Notese que $0 \in S$ por lo cual S es no vacio asique en virtud de la Caracterizacion de Σ-enumerabilidad dada en la Guia 7, deberemos encontrar un programa $\mathcal{P} \in \text{Pro}^{\Sigma}$ que enumere a S, es decir tal que $\text{Dom}\Psi^{1,0,\#}_{\mathcal{P}} = \omega$ y $\text{Im}\,\Psi^{1,0,\#}_{\mathcal{P}} = S$. Dicho en palabras, el programa $\mathcal{P}$ debera cumplir:

- Siempre que lo corramos desde un estado de la forma $\|x\|$, con $x \in \omega$, debe detenerse y el contenido de la variable N1 bajo detencion debera ser un elemento de S
- Para cada $s \in S$ debera haber un $x \in \omega$ tal que s es el valor de la variable N1 bajo detencion cuando corremos $\mathcal{P}$ desde $\|x\|$

Sea $\mathcal{P}_0 \in \text{Pro}^{\Sigma}$ un programa que compute a f. Usaremos $\mathcal{P}_0$ para diseñar $\mathcal{P}$. A continuacion daremos una descripcion intuitiva del funcionamiento de $\mathcal{P}$ (pseudocodigo) para luego escribirlo correctamente usando macros. El programa $\mathcal{P}$ comenzara del estado $\|x\|$ y hara las siguientes tareas

Etapa 1: si $x = 0$ ir a Etapa 6, en caso contrario ir a Etapa 2.
Etapa 2: calcular $(x)_1$ y $(x)_2$ e ir a Etapa 3.
Etapa 3: si $\mathcal{P}_0$ termina desde $\|(x)_1\|$ en $(x)_2$ pasos ir a Etapa 4, en caso contrario ir a Etapa 6
Etapa 4: si el valor que queda en N1 luego de correr $\mathcal{P}_0$ una cantidad $(x)_2$ de pasos, partiendo de $\|(x)_1\|$, es distinto de 0, entonces ir a Etapa 5. En caso contrario ir a Etapa 6.
Etapa 5: asignar a N1 el valor $(x)_1$ y terminar
Etapa 6: asignar a N1 el valor 0 y terminar

Notese que la descripcion anterior no es ni mas ni menos que un procedimiento efectivo (efectivisable) que enumera a S, y nuestra mision es simularlo dentro del lenguaje $\mathcal{S}^{\Sigma}$. Para esto usaremos varios macros. Ya que la funcion $f = \lambda x[(x)_1]$ es Σ-p.r., el Segundo Manantial de Macros nos dice que en $\mathcal{S}^{\Sigma}$ hay un macro:

$$[\text{V2} \leftarrow f(\text{V1})]$$

el cual escribiremos de la siguiente manera mas intuitiva:

$$[\text{V2} \leftarrow (\text{V1})_1]$$

Similarmente hay un macro:

$$[\text{V2} \leftarrow (\text{V1})_2]$$

Tambien, ya que el predicado $P = \lambda x[x = 0]$ es Σ-recursivo, hay un macro:

$$[\text{IF } P(\text{V1}) \text{ GOTO A1}]$$

el cual escribiremos de la siguiente manera:

$$[\text{IF V1} = 0 \text{ GOTO A1}]$$

Definamos

$$H = \lambda tx \left[Halt^{1,0}(t, x, \mathcal{P}_0)\right]$$

Notar que $D_H = \omega^2$ y que H es Σ-mixta. Ademas sabemos que la funcion $Halt^{1,0}$ es $(\Sigma \cup \Sigma_p)$-p.r. por lo cual resulta facilmente que H es $(\Sigma \cup \Sigma_p)$-p.r.. Por la Proposicion de Independencia del Alfabeto tenemos que H es Σ-p.r.. O sea que el Segundo Manantial de Macros nos dice que en $\mathcal{S}^{\Sigma}$ hay un macro:

$$[\text{IF } H(\text{V1}, \text{V2}) \text{ GOTO A1}]$$

Para hacer mas intuitivo el uso de este macro lo escribiremos de la siguiente manera

$$\left[\text{IF } Halt^{1,0}(\text{V1}, \text{V2}, \mathcal{P}_0) \text{ GOTO A1}\right]$$

Sea

$$g = \lambda tx \left[E^{1,0}_{\#1}(t, x, \mathcal{P}_0)\right]$$

Ya que g es Σ-recursiva (por que?), hay en $\mathcal{S}^{\Sigma}$ un macro:

$$[\text{V3} \leftarrow g(\text{V1}, \text{V2})]$$

Para hacer mas intuitivo el uso de este macro lo escribiremos de la siguiente manera

$$\left[\text{V3} \leftarrow E^{1,0}_{\#1}(\text{V1}, \text{V2}, \mathcal{P}_0)\right]$$

Ahora si podemos dar nuestro progama $\mathcal{P}$ que enumera a S:

```
    IF N1 ≠ 0 GOTO L1
    GOTO L2
L1  [N3 ← (N1)_1]
    [N4 ← (N1)_2]
    [IF Halt^{1,0}(N4, N3, P_0) GOTO L3]
    GOTO L2
L3  [N5 ← E^{1,0}_{#1}(N4, N3, P_0)]
    [IF N5 = 0 GOTO L2]
    N1 ← N3
    GOTO L4
L2  N1 ← 0
L4  SKIP
```

Para hacer la proxima tanda de ejercicios recomendamos al lector que repase la definicion de cuando "un programa $\mathcal{P} \in \text{Pro}^{\Sigma}$ enumera a un conjunto $S \subseteq \omega^n \times \Sigma^{*m}$" (ver despues de la Caracterizacion de Σ-enumerabilidad dada en la Guia 7).

Ejercicio 8: Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : D_f \subseteq \omega \times \omega \times \Sigma^* \to \omega$ es Σ-computable y $(0, 0, \varepsilon) \in D_f$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto D_f.

Ejercicio 9: Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : D_f \subseteq \omega \times \omega \times \Sigma^* \to \omega$ es Σ-computable y $(0, 0, \varepsilon) \in D_f$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto $\text{Im}(f)$.

Ejercicio 10: Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : L_1 \to L_2$, con $L_1, L_2 \subseteq \Sigma^*$, es Σ-computable y biyectiva. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que compute a f^{-1}.

Ejercicio 11: Sea $\Sigma = \{\#, \$\}$ y sea $f : D_f \subseteq \Sigma^* \to \omega$ una funcion Σ-computable tal que $f(\varepsilon) = 1$. Sea $L = \{\alpha \in D_f : f(\alpha) = 1\}$. De un programa $\mathcal{Q} \in \text{Pro}^{\Sigma}$ (usando macros) el cual enumere a L (explicado en video en granlogico.com).

Ejercicio 12: Sea $\Sigma = \{@, \%\}$ y sea $\mathcal{P}_0 \in \text{Pro}^{\Sigma}$. Sea

$$S = \{(x, \alpha) : \Psi^{1,0,\#}_{\mathcal{P}_0}(x) = \Psi^{0,1,\#}_{\mathcal{P}_0}(\alpha)\}$$

Note que $(0, \varepsilon) \in S$. De un programa $\mathcal{Q} \in \text{Pro}^{\Sigma}$ (usando macros) que enumere al conjunto S.

Ejercicio 13: Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : S \subseteq \omega \to \omega$ es Σ-computable, $0 \in S$ y $f(0) = 0$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto

$$H = \{x \in S : x \text{ es par } x/2 \in S \text{ y } f(x) = f(x/2)\}$$

Ejercicio 14: Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : S \subseteq \omega \to \omega$ es Σ-computable, $0 \in S$ y $f(0) = 0$. De un programa $\mathcal{P}_0 \in \text{Pro}^{\Sigma}$ (usando macros) tal que

$$\text{Dom}(\Psi^{1,0,\#}_{\mathcal{P}_0}) = \{x \in S : x \text{ es par } x/2 \in S \text{ y } f(x) = f(x/2)\}$$

Ejercicio 15: Sea $\Sigma = \{@, \&\}$. Supongamos $f : S \subseteq \omega \times \omega \to \omega$ es Σ-computable. Suponga ademas que $(0, 0) \in S$ y $f(0, 0) = 0$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto

$$H = \{(x, y) \in S : (f(x, y), y) \in S \text{ y } f(f(x, y), y) = 0\}$$

Ejercicio 16: Sean $f : D_f \subseteq \omega \times \Sigma^* \to \omega$ y $g : D_g \subseteq \omega \times \Sigma^* \to \omega$ funciones Σ-computables tales que $D_f \cap D_g = \emptyset$. Sea $F : D_f \cup D_g \to \omega$ dada por

$$e \quad \to \quad \begin{cases} f(e) & \text{si } e \in D_f \\ g(e) & \text{si } e \in D_g \end{cases}$$

De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que compute a F.

Enumeracion de conjuntos de programas. Ya que los programas de $\mathcal{S}^{\Sigma}$ son palabras del alfabeto $\Sigma \cup \Sigma_p$, nos podemos preguntar cuando un conjunto L de programas es $(\Sigma \cup \Sigma_p)$-enumerable. Daremos un ejemplo. Sea $\Sigma = \{@, !\}$ y sea

$$L = \{\mathcal{P} \in \text{Pro}^{\Sigma} : \Psi^{1,0,\#}_{\mathcal{P}}(10) = 10\}$$

Veremos que L es $(\Sigma \cup \Sigma_p)$-enumerable, dando un programa $\mathcal{Q} \in \text{Pro}^{\Sigma \cup \Sigma_p}$ que enumere a L, es decir tal que $\text{Dom}(\Psi^{1,0,*}_{\mathcal{Q}}) = \omega$ y $\text{Im}(\Psi^{1,0,*}_{\mathcal{Q}}) = L$. Cabe destacar

que aqui hay en juego dos versiones de nuestro lenguaje imperativo, es decir enumeraremos un conjunto de programas de $\mathcal{S}^{\Sigma}$ usando un programa de $\mathcal{S}^{\Sigma\cup\Sigma_p}$. Sea $\leq$ un orden total sobre el conjunto $\Sigma \cup \Sigma_p$.

A continuacion daremos una descripcion intuitiva del funcionamiento de $\mathcal{Q}$ (pseudocodigo) para luego escribirlo correctamente usando macros. Notese que SKIP $\in L$. El programa $\mathcal{Q}$ comenzara del estado $\|x\|$ y hara las siguientes tareas

> Etapa 1: si $x = 0$ ir a Etapa 6, en caso contrario ir a Etapa 2.
> Etapa 2: calcular $(x)_1$, $(x)_2$ y $*^{\leq}((x)_1)$ e ir a Etapa 3.
> Etapa 3: si $*^{\leq}((x)_1) \in \text{Pro}^{\Sigma}$ y termina partiendo desde $\|10\|$ en $(x)_2$ pasos ir a Etapa 4, en caso contrario ir a Etapa 6
> Etapa 4: si el valor que queda en N1 luego de correr $*^{\leq}((x)_1)$ una cantidad $(x)_2$ de pasos, partiendo de $\|10\|$ es igual a 10, entonces ir a Etapa 5. En caso contrario ir a Etapa 6.
> Etapa 5: asignar a P1 la palabra $*^{\leq}((x)_1)$ y terminar
> Etapa 6: asignar a P1 la palabra SKIP y terminar

Notese que la descripcion anterior no es ni mas ni menos que un procedimiento efectivo que enumera a L, y nuestra mision es simularlo dentro del lenguaje $\mathcal{S}^{\Sigma\cup\Sigma_p}$. Para esto usaremos varios macros. Es importante notar que los macros que usaremos corresponden al lenguaje $\mathcal{S}^{\Sigma\cup\Sigma_p}$ ya que los usaremos en $\mathcal{Q}$ el cual sera un programa de $\mathcal{S}^{\Sigma\cup\Sigma_p}$.

Ya que las funciones $\lambda x[(x)_1]$ y $\lambda x[(x)_2]$ son $(\Sigma \cup \Sigma_p)$-recursivas el Segundo Manantial nos dice que hay macros en $\mathcal{S}^{\Sigma\cup\Sigma_p}$ asociados a estas funciones los cuales escribiremos de la siguiente manera mas intuitiva:

$$[\text{V2} \leftarrow (\text{V1})_1]$$
$$[\text{V2} \leftarrow (\text{V1})_2]$$

Ya que el predicado $P = \lambda x[x = 10]$ es $(\Sigma \cup \Sigma_p)$-recursivo tenemos en $\mathcal{S}^{\Sigma\cup\Sigma_p}$ su macro asociado el cual escribiremos de la siguiente manera:

$$[\text{IF V1} = 10 \text{ GOTO A1}]$$

Por un lema anterior sabemos que Pro^{Σ} es un conjunto $(\Sigma \cup \Sigma_p)$-p.r., por lo cual $\chi_{\text{Pro}^{\Sigma}}^{(\Sigma\cup\Sigma_p)^*}$ es $(\Sigma \cup \Sigma_p)$-p.r., y por lo tanto hay un macro

$$\left[\text{IF } \chi_{\text{Pro}^{\Sigma}}^{(\Sigma\cup\Sigma_p)^*}(\text{W1}) \text{ GOTO A1}\right]$$

el cual escribiremos de la siguiente manera

$$\left[\text{IF W1} \in \text{Pro}^{\Sigma} \text{ GOTO A1}\right]$$

Ya que el predicado $Halt^{1,0}$ es $(\Sigma \cup \Sigma_p)$-recursivo tenemos un macro asociado a el, el cual escribiremos de la siguiente forma

$$\left[\text{IF } Halt^{1,0}(\text{V1}, \text{V2}, \text{W1}) \text{ GOTO A1}\right]$$

Ya que $E_{\#1}^{1,0}$ es $(\Sigma \cup \Sigma_p)$-recursivo tenemos un macro asociado a ella, el cual escribiremos de la siguiente forma

$$\left[\text{V3} \leftarrow E_{\#1}^{1,0}(\text{V1}, \text{V2}, \text{W1})\right]$$

Tambien usaremos macros

$$[\text{V1} \leftarrow 10]$$
$$[\text{W1} \leftarrow \text{SKIP}]$$

(dejamos al lector hacerlos a mano o tambien se puede justificar su existencia via el Segundo Manantial aplicado a las funciones $C^{0,0}_{10}$ y $C^{0,0}_{\text{SKIP}}$).

Ahora si podemos hacer el programa $\mathcal{Q}$ de $\mathcal{S}^{\Sigma\cup\Sigma_p}$ que enumera a L:

	IF N1 $\neq 0$ GOTO L1
	GOTO L2
L1	$[\text{N2} \leftarrow (\text{N1})_1]$
	$[\text{N3} \leftarrow (\text{N1})_2]$
	$[\text{P1} \leftarrow *^{\leq}(\text{N2})]$
	$\left[\text{IF P1} \in \text{Pro}^{\Sigma} \text{ GOTO L3}\right]$
	GOTO L2
L3	$[\text{N4} \leftarrow 10]$
	$\left[\text{IF } Halt^{1,0}(\text{N3}, \text{N4}, \text{P1}) \text{ GOTO L4}\right]$
	GOTO L2
L4	$\left[\text{N5} \leftarrow E^{1,0}_{\#1}(\text{N3}, \text{N4}, \text{P1})\right]$
	[IF N5 = 10 GOTO L5]
L2	[P1 $\leftarrow$ SKIP]
L5	SKIP

Ejercicio 17: (Explicado en video en granlogico.com.) Sea $\Sigma = \{\#, \$\}$ y sea

$$L = \{\mathcal{P} \in \text{Pro}^{\Sigma} : \exists n, \alpha \text{ tales que } \Psi^{2,1,*}_{\mathcal{P}}(|\mathcal{P}|, n, \alpha) = \$\}$$

(a) Dar un programa $\mathcal{Q} \in \text{Pro}^{\Sigma\cup\Sigma_p}$ tal que $\text{Dom}(\Psi^{0,1,\#}_{\mathcal{Q}}) = L$

(b) Dar un programa $\mathcal{Q}' \in \text{Pro}^{\Sigma\cup\Sigma_p}$ tal que $\text{Dom}(\Psi^{1,0,*}_{\mathcal{Q}'}) = \omega$ y $\text{Im}(\Psi^{1,0,*}_{\mathcal{Q}'}) = L$

Para cada macro que use dar la funcion o predicado asociado y justificar por que existe tal macro.

Ejercicio 18: Sea $\Sigma = \{@, !\}$ y sea

$$L = \{\mathcal{P} \in \text{Pro}^{\Sigma} : \exists \alpha \text{ tal que } \Psi^{0,2,*}_{\mathcal{P}}(\alpha, \alpha\alpha) = \alpha\alpha\alpha\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma\cup\Sigma_p}$ el cual enumere a L. Para cada macro que use dar la funcion o predicado asociado y justificar por que existe tal macro.

Ejercicio 19: Sea $\Sigma = \{@, \%\}$ y sea

$$S = \{(x, \mathcal{P}) \in \omega \times \text{Pro}^{\Sigma} : x \in \text{Dom}(\Psi^{1,0,\#}_{\mathcal{P}})\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma\cup\Sigma_p}$ el cual enumere a S. Para cada macro que use dar la funcion o predicado asociado y justificar por que existe tal macro.

Ejercicio 20: Sea $\Sigma = \{@, \%\}$ y sea

$$S = \{(x, \mathcal{P}) \in \omega \times \text{Pro}^{\Sigma} : \Psi^{1,0,\#}_{\mathcal{P}}(x) = x^2\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma\cup\Sigma_p}$ el cual enumere a S. Para cada macro que use dar la funcion o predicado asociado y justificar por que existe tal macro.

Ejercicio 21: Sea $\Sigma = \{@, \%\}$ y sea

$$S = \{\mathcal{P}_1 \alpha \mathcal{P}_2 : \mathcal{P}_1, \mathcal{P}_2 \in \text{Pro}^{\Sigma}, \alpha \in \Sigma^* \text{ y } \Psi^{0,1,*}_{\mathcal{P}_1}(\alpha) = \Psi^{0,1,*}_{\mathcal{P}_2}(\alpha)\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma \cup \Sigma_p}$ el cual enumere a S. Para cada macro que use dar la funcion o predicado asociado y justificar por que existe tal macro.

Cuando $\Sigma \supseteq \Sigma_p$ la cosa se pone mas loca...

Cuando $\Sigma \supseteq \Sigma_p$ podemos correr un programa $\mathcal{P} \in \text{Pro}^{\Sigma}$ partiendo de un estado que asigne a sus variables alfabeticas programas (ya que los programas son meras palabras de Σ^*). En particular podriamos correr un programa $\mathcal{P}$ desde el estado $\|\mathcal{P}\|$. Llamaremos A al conjunto formado por aquellos programas $\mathcal{P}$ tales que $\mathcal{P}$ se detiene partiendo del estado $\|\mathcal{P}\|$. Es decir

$$A = \{\mathcal{P} \in \text{Pro}^{\Sigma} : \exists t \in \omega \text{ tal que } Halt^{0,1}(t, \mathcal{P}, \mathcal{P}) = 1\}$$

Por ejemplo SKIP $\in A$. Dicho rapida y sugestivamente A es el conjunto formado por aquellos programas que se detienen partiendo de si mismos. Dejamos como ejercicio la tarea de hacer un programa que enumere a A. Como veremos en la Guia 9, el conjunto A, si bien es Σ-enumerable, no es Σ-computable.

Ahora consiremos el conjunto

$$L = \{\mathcal{P} \in \text{Pro}^{\Sigma} : \Psi^{0,0,*}_{\mathcal{P}}(\Diamond) = \mathcal{P}\}$$

En palabras, un programa $\mathcal{P} \in \text{Pro}^{\Sigma}$ estara en L si y solo si $\mathcal{P}$ termina desde $((0,0,...),(\varepsilon,\varepsilon,...))$ dejando en P1 la palabra $\mathcal{P}$. En algun sentido los elementos de L son programas que se autopropagandean ya que "de la nada ellos terminan dandose a si mismos como salida". Dejamos al lector la tarea de reflexionar hasta entender que resulta dificil encontrar "a mano" un elemento de L. En algun sentido, para lograr hacer un programa que este en L debemos ir escribiendo instrucciones que a su ves vayan garantizando que ellas mismas vayan quedando en el contenido de la variable P1. Sin embargo el Teorema de la Recursion (de Kleene) nos garantiza que L es no vacio! Dejamos como ejercicio la tarea de hacer un programa que enumere a L (obviamente utilizando un elemento fijo de L).

Consideremos ahora el conjunto

$$S = \{(\mathcal{P}_1, \mathcal{P}_2) \in \text{Pro}^{\Sigma} \times \text{Pro}^{\Sigma} : \Psi^{0,0,*}_{\mathcal{P}_1}(\Diamond) = \mathcal{P}_1\mathcal{P}_2 = \Psi^{0,0,*}_{\mathcal{P}_2}(\Diamond)\}$$

Note que nuevamente es dificil imaginar como uno programaria un par de programas $\mathcal{P}_1$ y $\mathcal{P}_2$ de manera que $(\mathcal{P}_1, \mathcal{P}_2)$ este en S. El Teorema de la Recursion Doble (de Smullyan) nos garantiza que S es no vacio! Dejamos al lector la tarea de hacer un programa que enumere a L (obviamente utilizando un elemento fijo de S).

Ejercicio 22: Supongamos que $\Sigma \supseteq \Sigma_p$. De un programa de $\mathcal{S}^{\Sigma}$ que enumere a A.

Ejercicio 23: Supongamos que $\Sigma \supseteq \Sigma_p$. Sea

$$L = \{\mathcal{P} \in \text{Pro}^{\Sigma} : \Psi^{0,0,*}_{\mathcal{P}}(\Diamond) = \mathcal{P}\}$$

De un programa de $\mathcal{S}^{\Sigma}$ que enumere a L.

Ejercicio 24: Supongamos que $\Sigma \supseteq \Sigma_p$. Sea

$$L = \{\mathcal{P} \in \text{Pro}^{\Sigma} : \Psi^{2,1,*}_{\mathcal{P}}(1,1,\mathcal{P}) = \mathcal{P}\mathcal{P}\}$$

Encuentre un elemento concreto de L y de un programa de $\mathcal{S}^{\Sigma}$ que enumere a L

Ejercicio 25: Supongamos que $\Sigma \supseteq \Sigma_p$. De un programa de $\mathcal{S}^\Sigma$ que enumere al conjunto

$$L = \{\mathcal{P} \in \text{Pro}^\Sigma : \exists x \text{ tal que } \Psi_{\mathcal{P}}^{1,1,*}(x, \mathcal{P}) = I_1^{\mathcal{P}}\}$$

Ejercicio 26: Supongamos que $\Sigma \supseteq \Sigma_p$. De un programa de $\mathcal{S}^\Sigma$ que enumere al conjunto

$$S = \{(x, \mathcal{P}, \alpha) \in \omega \times \text{Pro}^\Sigma \times \Sigma^* : \Psi_{\mathcal{P}}^{1,2,\#}(x, \alpha, \mathcal{P}^x) = 0\}$$

Ejercicio 27: Supongamos que $\Sigma \supseteq \Sigma_p$. Sea

$$S = \{(\mathcal{P}_1, \mathcal{P}_2) \in \text{Pro}^\Sigma \times \text{Pro}^\Sigma : \Psi_{\mathcal{P}_1}^{0,0,*}(\Diamond) = \mathcal{P}_1\mathcal{P}_2 = \Psi_{\mathcal{P}_2}^{0,0,*}(\Diamond)\}$$

De un programa de $\mathcal{S}^\Sigma$ que enumere al conjunto S

Dos batallas mas

Aceptaremos sin prueba el siguiente teorema.

Teorema 12 (Godel vence a Turing)**.** *Supongamos $f : S \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-Turing computable. Entonces f es Σ-recursiva.*

Ejercicio 28: Haga un esquema a nivel de ideas (sin demostraciones y sin demaciada precision) de la prueba del teorema anterior (la prueba completa esta en el apunte).

Aceptaremos sin prueba el siguiente teorema.

Teorema 13 (Turing vence a Neumann)**.** *Si $f : D_f \subseteq \omega^n \times \Sigma^{*m} \to O$ es Σ-computable, entonces f es Σ-Turing computable.*

Ejercicio 29: Haga un esquema a nivel de ideas (sin demostraciones y sin demaciada precision) de la prueba del teorema anterior (la prueba completa esta en el apunte y en granlogico.com hay un video con la prueba del teorema)

La tesis de Church

En virtud de los teoremas ya expuestos tenemos el siguiente teorema que asegura que los tres paradigmas son equivalentes.

Teorema 14. *Sea Σ un alfabeto finito. Dada una funcion f, las siguientes son equivalentes:*

(1) *f es Σ-Turing computable*
(2) *f es Σ-recursiva*
(3) *f es Σ-computable.*

Tambien los tres paradigmas son equivalentes con respecto a los dos tipos de conjuntos estudiados, es decir:

Teorema 15. *Sea Σ un alfabeto finito y sea $S \subseteq \omega^n \times \Sigma^{*m}$. Las siguientes son equivalentes:*

(1) *S es Σ-Turing enumerable*
(2) *S es Σ-recursivamente enumerable*
(3) *S es Σ-enumerable*

Teorema 16. *Sea Σ un alfabeto finito y sea $S \subseteq \omega^n \times \Sigma^{*m}$. Las siguientes son equivalentes:*

(1) *S es Σ-Turing computable*
(2) *S es Σ-recursivo*
(3) *S es Σ-computable*

Ejercicio 30: Pruebe los dos teoremas anteriores

Otro modelo matematico de computabilidad efectiva es el llamado lambda calculus, introducido por Church, el cual tambien resulta equivalente a los estudiados por nosotros. El hecho de que tan distintos paradigmas computacionales hayan resultado equivalentes hace pensar que en realidad los mismos han tenido exito en capturar la totalidad de las funciones Σ-efectivamente computables. Esta aceveracion es conocida como la

Tesis de Church: *Toda funcion Σ-efectivamente computable es Σ-recursiva.*

Y por supuesto puede ser sintetizada diciendo que Godel vence a Leibniz (y por lo tanto los cuatro proceres empatan!). Si bien no se ha podido dar una prueba estrictamente matematica de la Tesis de Church, es un sentimiento comun de los investigadores del area que la misma es verdadera.

Ejercicios de examen

Para cada macro que use dar la funcion o predicado asociado y justificar por que existe tal macro usando alguno de los dos Manantiales de Macros (el primer Manantial esta en la Guia 7 y el segundo en esta).

(1) Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : S \subseteq \omega \times \omega \times \Sigma^* \to \omega$ es Σ-computable. Suponga ademas que $(0, 0, \varepsilon) \in S$ y $f(0, 0, \varepsilon) = 0$. De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto

$$H = \{(x, y, \alpha) \in S : f(x, y, \alpha) = 0\}$$

(2) Sea $\Sigma = \{@, \&\}$. Supongamos $f : S \subseteq \omega \times \omega \to \omega$ es Σ-computable. Suponga ademas que $(0, 0) \in S$ y $f(0, 0) = 0$. De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto

$$H = \{(x, y) \in S : f(x, y) = x\}$$

(3) Sea $\Sigma = \{@, \&\}$. Supongamos $f : L \subseteq \Sigma^* \times \Sigma^* \to \Sigma^*$ es Σ-computable. Suponga ademas que $(@, \&) \in L$ y $f(@, \&) = @@@$. De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto

$$H = \{(\alpha, \beta) \in L : f(\alpha, \beta) = \alpha\alpha\alpha\}$$

(4) Sea $\Sigma = \{@, \&\}$. Supongamos $f : S \subseteq \omega \times \omega \to \omega$ es Σ-computable. Suponga ademas que $(0, 0) \in S$ y $f(0, 0) = 0$. De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto

$$H = \{(x, y) \in S : (y, x) \in S \text{ y } f(x, y) = f(y, x)\}$$

(5) Sea $\Sigma = \{@, \&\}$. Supongamos $f : S \subseteq \omega \times \omega \to \omega$ es Σ-computable. Suponga ademas que $(0,0) \in S$ y $f(0,0) = 0$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto

$$H = \{(x,y) \in S : (f(x,y),y) \in S \text{ y } f(f(x,y),y) = 0\}$$

(6) Sea $\Sigma = \{@, \&\}$. Supongamos $f : S \subseteq \omega \times \omega \to \omega$ es Σ-computable. Suponga ademas que $(0,0) \in S$ y $f(0,0) = 0$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto

$$H = \{x \in \omega : \text{ hay un } y\text{tal que } (x,y) \in S \text{ y } f(x,y) = 0\}$$

(7) Sea $\Sigma = \{@, \&\}$. Supongamos $f : S \subseteq \omega \times \omega \to \omega$ es Σ-computable. Suponga ademas que $(0,0) \in S$ y $f(0,0) = 0$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto

$$H = \{(x,x) : (x,x) \in S \text{ y } f(x,x) = x\}$$

(8) Sea $\Sigma = \{@, \&\}$. Supongamos $f : L \subseteq \Sigma^* \times \Sigma^* \to \Sigma^*$ es Σ-computable. Suponga ademas que $(@, @) \in L$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto

$$H = \{(\alpha,\beta) \in L : (\beta,\alpha) \in L \text{ y } f(\alpha,\beta) = f(\beta,\alpha)\}$$

(9) Sea $\Sigma = \{@, \&\}$. Sea $f : S \subseteq \Sigma^* \to \omega$ una funcion Σ-computable. Supongamos que $\varepsilon \in S$ y que $f(\varepsilon) = 0$. Notar que $\varepsilon \in L$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto

$$L = \{\alpha \in S : @^{f(\alpha)} \in S \text{ y } f(@^{f(\alpha)}) = 0\}$$

(10) Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : S \subseteq \omega \times \omega \times \Sigma^* \to \omega$ es Σ-computable. Suponga ademas que $(0,0,\varepsilon) \in S$ y $f(0,0,\varepsilon) = 0$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto

$$H = \{(x,y,\alpha) \in S : (y,x,\alpha) \in S \text{ y } f(x,y,\alpha) = f(y,x,\alpha)\}$$

(11) Sea $\Sigma = \{@, \%\}$ y sea $\mathcal{P}_0 \in \text{Pro}^{\Sigma}$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto

$$L = \{\alpha \in \Sigma^* : (\exists x \in \mathbf{N})\ \Psi^{1,1,\#}_{\mathcal{P}_0}(x^2,\alpha) = \Psi^{0,2,\#}_{\mathcal{P}_0}(\alpha,\alpha)\}$$

(note que $\varepsilon \in L$).

(12) Sea $\Sigma = \{@, \%\}$ y sea $\mathcal{P}_0 \in \text{Pro}^{\Sigma}$. Sea

$$S = \{(x,\alpha) : \Psi^{1,0,\#}_{\mathcal{P}_0}(x) = \Psi^{0,1,\#}_{\mathcal{P}_0}(\alpha)\}$$

Note que $(0,\varepsilon) \in S$. De un programa $\mathcal{Q} \in \text{Pro}^{\Sigma}$ (usando macros) que enumere al conjunto S.

(13) Sea $\Sigma = \{@, !, \%\}$. Supongamos $f_1 : S_1 \subseteq \omega \times \omega \times \Sigma^* \to \omega$ y $f_2 : S_2 \subseteq \omega \times \omega \times \Sigma^* \to \Sigma^*$ son Σ-computables y $(0,0,\varepsilon) \in S_1 \cap S_2$. De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto $\text{Im}([f_1, f_2])$.

(14) Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : S \subseteq \omega \to \omega$ y $g : L \subseteq \Sigma^* \to \Sigma^*$ son Σ-computables. Supongamos ademas que 0 pertenece al conjunto

$$H = \{x : x \in S,\ @^x \in L \text{ y } f(x) = g(@^x)\}$$

De un programa de $\mathcal{S}^{\Sigma}$ (usando macros) que enumere al conjunto H.

(15) Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : S \subseteq \omega \to \omega$ y $g : L \subseteq \Sigma^* \to \omega$ son Σ-computables. Supongamos ademas que ε pertenece al conjunto

$$H = \{\alpha \in L : g(\alpha) \in S \text{ y } f(g(\alpha)) = 0\}$$

De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto H.

(16) Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : S \subseteq \omega \to \omega$ es Σ-computable, $0 \in S$ y $f(0) = 0$. De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto

$$H = \{x \in S : x^2 \in S \text{ y } f(x) = f(x^2)\}$$

(17) Sea $\Sigma = \{@, !, \%\}$. Supongamos $f : S \subseteq \omega \to \omega$ es Σ-computable, $0 \in S$ y $f(0) = 0$. De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto

$$H = \{x \in S : x \text{ es par } x/2 \in S \text{ y } f(x) = f(x/2)\}$$

(18) Sea $\Sigma = \{@, !, \%\}$. Supongamos $L_1, L_2 \subseteq \Sigma^*$ son Σ-enumerables. Supongamos ademas que ε pertenece al conjunto

$$H = \{\alpha \in L_1 : \text{hay un } \beta \in L_2 \text{ tal que } |\beta| = |\alpha|\}$$

De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto H.

(19) Sea $\Sigma = \{!, \%\}$. Supongamos $S \subseteq \omega \times \omega \times \Sigma^*$ y $L \subseteq \Sigma^*$ son Σ-enumerables. Supongamos ademas que $(0, 0, \varepsilon)$ pertenece al conjunto

$$H = \{(x, y, \alpha) \in S : \text{hay un } \beta \in L \text{ tal que } |\beta| = |\alpha|\}$$

De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto H.

(20) Sea $\Sigma = \{!, \%\}$. Supongamos $S \subseteq \omega \times \omega$ y $S' \subseteq \omega$ son Σ-enumerables. Supongamos ademas que $(0, 0)$ pertenece al conjunto

$$H = \{(x, y) : (x, y) \in S \text{ y } x + y \in S'\}$$

De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto H.

(21) Sea $\Sigma = \{!, \%\}$. Supongamos $S \subseteq \Sigma^* \times \Sigma^*$ y $L \subseteq \Sigma^*$ son Σ-enumerables. Supongamos ademas que $(\varepsilon, \varepsilon)$ pertenece al conjunto

$$H = \{(\alpha, \beta) \in S : \text{hay un } \gamma \in L \text{ tal que } \alpha = \beta\gamma\}$$

De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto H.

(22) Sean $f : D_f \subseteq \omega \times \Sigma^* \to \omega$ y $g : D_g \subseteq \omega \times \Sigma^* \to \omega$ funciones Σ-computables tales que $D_f \cap D_g = \emptyset$. Sea $F : D_f \cup D_g \to \omega$ dada por

$$e \quad \to \quad \begin{cases} f(e) & \text{si } e \in D_f \\ g(e) & \text{si } e \in D_g \end{cases}$$

De un programa de $\mathcal{S}^\Sigma$ (usando macros) que compute a F.

(23) Sea $\Sigma = \{@, !, \%\}$. Supongamos $L_1, L_2 \subseteq \Sigma^*$ son Σ-enumerables. Supongamos ademas que ε pertenece al conjunto

$$H = \{\alpha \in L_1 : \text{hay un } \beta \in L_2 \text{ tal que } \beta^3 = \alpha\}$$

De un programa de $\mathcal{S}^\Sigma$ (usando macros) que enumere al conjunto H.

(24) Sea $\Sigma = \{@, \%\}$ y sea

$$S = \{(x, \mathcal{P}, \alpha) \in \omega \times \text{Pro}^\Sigma \times \Sigma^* : \Psi_{\mathcal{P}}^{1,1,\#}(x, \alpha) = 0\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma \cup \Sigma_p}$ el cual enumere a S.

(25) Sea $\Sigma = \{@, \%\}$ y sea

$$S = \{(\mathcal{P}_1, \mathcal{P}_2) \in \text{Pro}^\Sigma \times \text{Pro}^\Sigma : \exists \alpha \text{ tal que } \Psi^{0,1,*}_{\mathcal{P}_1}(\alpha) = \Psi^{0,1,*}_{\mathcal{P}_2}(\alpha)\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma \cup \Sigma_p}$ el cual enumere a S.

(26) Sea $\Sigma = \{@, !\}$ y sea

$$L = \{\mathcal{P} \in \text{Pro}^\Sigma : \exists \alpha \text{ tal que } \Psi^{1,1,*}_{\mathcal{P}}(11, \alpha) = \alpha!@\}$$

(a) Dar un programa $\mathcal{Q} \in \text{Pro}^{\Sigma \cup \Sigma_p}$ tal que $\text{Dom}(\Psi^{0,1,\#}_{\mathcal{Q}}) = L$

(b) Dar un programa $\mathcal{Q}' \in \text{Pro}^{\Sigma \cup \Sigma_p}$ tal que $\text{Dom}(\Psi^{1,0,*}_{\mathcal{Q}'}) = \omega$ y $\text{Im}(\Psi^{1,0,*}_{\mathcal{Q}'}) = L$

(27) Sea $\Sigma = \{@, !\}$ y sea

$$L = \{\mathcal{P} \in \text{Pro}^\Sigma : \exists \alpha \text{ tal que } \Psi^{0,2,*}_{\mathcal{P}}(\alpha, \alpha\alpha) = \alpha\alpha\alpha\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma \cup \Sigma_p}$ el cual enumere a L.

(28) Sea $\Sigma = \{@, \%\}$ y sea

$$S = \{(x, \mathcal{P}) \in \omega \times \text{Pro}^\Sigma : x \in \text{Dom}(\Psi^{1,0,\#}_{\mathcal{P}})\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma \cup \Sigma_p}$ el cual enumere a S.

(29) Sea $\Sigma = \{@, \%\}$ y sea

$$S = \{(x, \mathcal{P}) \in \omega \times \text{Pro}^\Sigma : \Psi^{1,0,\#}_{\mathcal{P}}(x) = x^2\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma \cup \Sigma_p}$ el cual enumere a S.

(30) Sea $\Sigma = \{@, \%\}$ y sea

$$S = \{\mathcal{P}_1 \alpha \mathcal{P}_2 : \mathcal{P}_1, \mathcal{P}_2 \in \text{Pro}^\Sigma, \alpha \in \Sigma^* \text{ y } \Psi^{0,1,*}_{\mathcal{P}_1}(\alpha) = \Psi^{0,1,*}_{\mathcal{P}_2}(\alpha)\}$$

Dar $\mathcal{Q} \in \text{Pro}^{\Sigma \cup \Sigma_p}$ el cual enumere a S.

(31) Supongamos que $\Sigma \supseteq \Sigma_p$. Sea

$$L = \{\mathcal{P} \in \text{Pro}^\Sigma : \Psi^{2,1,*}_{\mathcal{P}}(1, 1, \mathcal{P}) = \mathcal{P}\mathcal{P}\}$$

Encuentre un elemento concreto de L y de un programa de $\mathcal{S}^\Sigma$ que enumere a L

(32) Supongamos que $\Sigma \supseteq \Sigma_p$. De un programa de $\mathcal{S}^\Sigma$ que enumere al conjunto

$$L = \{\mathcal{P} \in \text{Pro}^\Sigma : \exists x \text{ tal que } \Psi^{1,1,*}_{\mathcal{P}}(x, \mathcal{P}) = I^{\mathcal{P}}_1\}$$

(33) Supongamos que $\Sigma \supseteq \Sigma_p$. De un programa de $\mathcal{S}^\Sigma$ que enumere al conjunto

$$S = \{(x, \mathcal{P}, \alpha) \in \omega \times \text{Pro}^\Sigma \times \Sigma^* : \Psi^{1,2,\#}_{\mathcal{P}}(x, \alpha, \mathcal{P}^x) = 0\}$$

(34) Supongamos que $\Sigma \supseteq \Sigma_p$. Sea

$$S = \{(\mathcal{P}_1, \mathcal{P}_2) \in \text{Pro}^\Sigma \times \text{Pro}^\Sigma : \mathcal{P}_1 \neq \mathcal{P}_2 \text{ y } \Psi^{0,1,*}_{\mathcal{P}_1}(\mathcal{P}_2) = \Psi^{0,1,*}_{\mathcal{P}_2}(\mathcal{P}_1)\}$$

Note que $(\text{P1} \leftarrow \varepsilon\text{SKIP}, \text{P1} \leftarrow \varepsilon) \in S$. De un programa de $\mathcal{S}^\Sigma$ que enumere al conjunto S